Lidar-based Hazard Avoidance for Safe Landing on Mars

Andrew Johnson, Allan Klumpp, James Collier and Aron Wolf
Jet Propulsion Laboratory, California Institute of Technology
Mail Stop 125-209, 4800 Oak Grove Drive, Pasadena, CA 91109

Abstract

One approach to safe landing on Mars is the hazard avoidance paradigm. During hazard avoidance the lander uses onboard sensors to detect hazards in the landing zone, autonomously selects a safe landing site, and then maneuvers to the new site. Design of a system for hazard avoidance is facilitated by simulation where trades involving sensor and mission requirements can be explored. This paper describes the algorithms and models that comprise a scanning lidar-based hazard avoidance simulation including a terrain generator, a lidar model, hazard avoidance algorithms and powered landing guidance algorithms.

1 Introduction

Safe landing on Mars can be achieved by either of two design approaches. One is "robustness", in which the spacecraft is designed to withstand impact with whatever terrain is expected in the landing zone; airbags (employed by Mars Pathfinder) are an example. This can be an unwieldy approach for large landers under consideration for future missions. The second approach (and the focus of this work) is "hazard avoidance" in which the spacecraft uses onboard sensors to detect hazards in the landing zone, selects an alternate landing site, and then maneuvers to the new site. Design of a system for hazard avoidance requires exploring trades involving sensor and mission requirements. Analysis of these systems trades is greatly facilitated by simulation. An integrated simulation tool has been developed and is described here, along with some illustrative results from a preliminary trade study. Our simulation contains four modules that interact according to the block diagram shown in Figure 1.

Terrain Generator: Topographic terrain data is needed to model the hazards (rocks, cliffs, craters) likely to be encountered during landing. A large database of high resolution Martian terrain is not available, so a method for synthetically generating realistic terrain is needed. Since the results generated from the simulation will be useful only if the underlying terrain is realistic, we use a method for generating Martian terrain that is based on geophysical processes. The terrain is generated by populating an initial surface, which comes from coarse orbiter topography data, with rocks and craters in a way that models the aging of the Martian surface. Once generated, the terrain is interrogated by the lidar model to generate measurements of surface topography. A reduced resolution of a nominal Martian terrain is shown in Figure 5 (A).

Lidar Model: A scanning lidar is currently the terrain sensing instrument base lined for the Mars 2007 Smart Lander mission, so it is the sensor we model in our simulation. A scanning lidar senses the 3-D topography within its field of view by raster scanning a pulsed laser beam across the targeted surface. By measuring the time of flight of the laser pulses reflected from the surface the range to the surface can be determined for each scan. When combined with measurements of the angular position of a mirror that directs the scan, a 3-D point or sample can be generated for each laser pulse. The output of the lidar is a cloud of 3-D points that convey the topography of the scanned surface. Assumed parameters for a landing lidar are a 10°x10° field of view with 10000 samples scanned in one second and a maximum range of 2km with a range resolution of 2cm. We have built a model of a lidar into our simulation that incorporates pointing errors and range sensing errors due to measurement noise as well as pulse stretching by the scanned terrain. The lidar model uses efficient ray tracing algorithms from computer graphics to generate 10000 samples in less than one second, so it can be used for real-time simulation of landing. The samples generated by the lidar model are output to the hazard detection and avoidance algorithms that compute safe landing sites.

Hazard Avoidance: A particular patch of terrain presents a hazard to the spacecraft during landing if the slope of the patch is too steep or the patch contains rocks or other protuberances that are taller than a certain terrain height. To quantitatively determine if a patch is hazardous, the slope and terrain variation over the patch must be measured. We have developed algorithms that estimate the location of surface hazards given scanning lidar data and incorporated them directly into the simulation. These algorithms build an elevation map from lidar samples, estimate local slope and roughness using the elevation map and then determine areas that exceed constraints on surface slope and roughness given the footprint of the lander. Images of nominal parameter maps are given in Figure 5 (B). Given all of the safe places to land within the field of view of the sensor, hazard avoidance algorithms pick the safe landing location that is farthest from all detected hazards and closest to the landing site selected during the previous scan. This new safe landing site is then passed to the powered landing guidance module that uses it to compute a trajectory to the new

landing site. A safe landing map with previous and new landing sites selected is shown in Figure 5 (C).

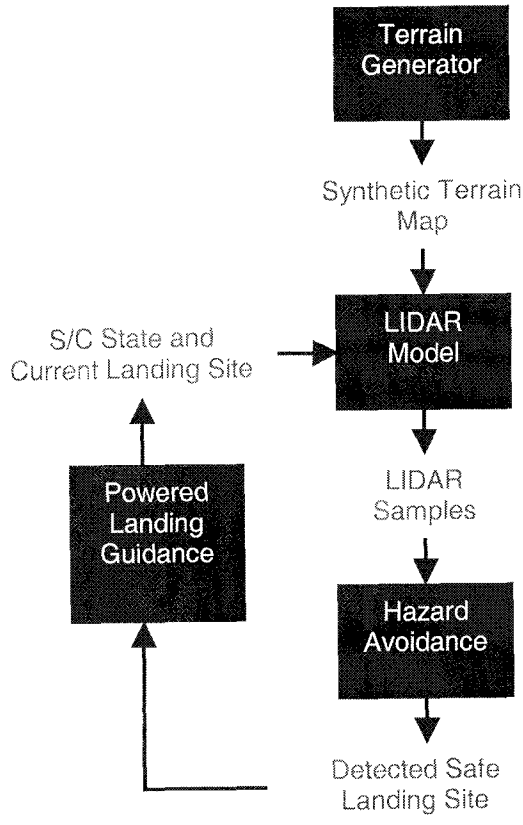

Figure 1 Block Diagram for simulation.

Powered Landing Guidance: We use an algorithm for powered-landing guidance that can retarget at any time to a new point specified by hazard avoidance. Our guidance algorithm, adapted from that flown on Apollo, transfers the lander from any current state (position and velocity) to touchdown in two phases, called "approach" and "vertical descent". Each phase has a target point where the position, velocity, and acceleration are all specified. The approach-phase target is five or ten meters directly above the landing site, and the vertical-descent-phase target is on the surface. Specifying zero for the horizontal components of target velocity and acceleration causes the lander to arrive at each target point with zero transverse velocity and in an erect attitude, regardless of any maneuvering en route to avoid hazards. The vertical components of approach-phase target velocity and acceleration are chosen to provide a fuel-efficient transfer with a safe thrust margin, and a vertical rate at the target that is the same or close to the constant value flown in the vertical-descent phase. In the vertical-descent phase, the lander descends at around 1 m/s until the engines are shut off upon or just before contacting the surface.

The simulation starts with the lander traveling along a predetermined trajectory. Each time a lidar scan is taken, hazards are detected and a (possibly) new safe landing site is selected. This site is then passed to the guidance module and it computes a new trajectory to the desired landing site. This process is repeated until the lander has landed safely on the surface.

The rest of the paper describes in detail the components of this simulation.

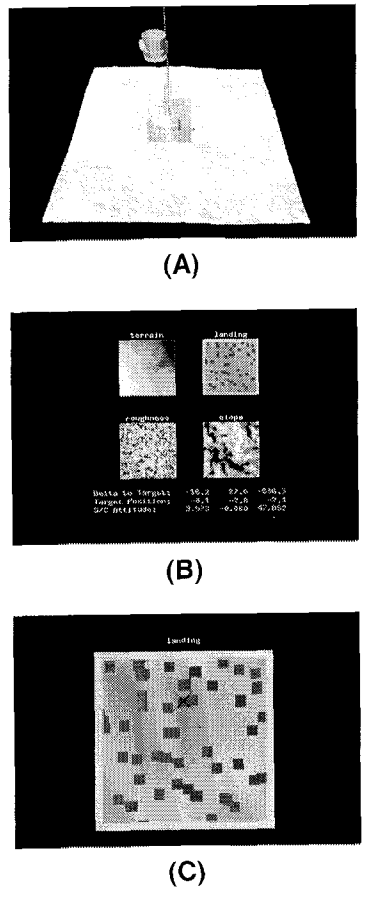

Figure 2 Hazard avoidance simulation visualization (A) 3-D visualization of terrain and lidar scan. (B) Parameter maps used to detect safe landing sites. (C) Safe landing site map with hazards shown in red, safe zones shown in green, previous landing sites shown as a black X and the safe landing site selected shown as a purple +.

2 Synthetic Terrain Map Generation

A requirement for any landing simulation is a physically accurate model of the surface being landed on. This model must contain enough fidelity that the sensor models that interact with the surface can produce measurements that are realistic. The simulation described in this paper uses a sophisticated environment for modeling Martian terrain that has been developed at JPL over the last decade[3][4].

The terrain generator software builds Martian surfaces by sequentially applying geological processes to the surface. These processes are realizations of the models used by planetary geologists to describe the surface of Mars. Of particular interest to safe landing is the statistical model for rock size and density. This process has one parameter that can be set by the user of the terrain generator.

The terrain generator begins with an initial coarse surface. Fractal surface generation is then used to fill in the terrain. This fractal surface is then acted upon by a cratering process and a rock generation process. These processes can be repeated to generate a surface that has the appearance of one that has been created over time. The main parameter of interest for simulation of safe landing is the rock density parameter.

The Martian surface terrain generator has been implemented on a 128 node SGI supercomputer. The interface to the terrain generator is a terrain server. The simulation requests a piece of terrain of a particular size, resolution and rock density from the terrain server. The terrain server then partitions the generation of the terrain into multiple parts and passes the compute intensive terrain generation off to multiple processors on the supercomputer. The terrain pieces are generated and the terrain server puts them together and sends the complete terrain to the simulation.

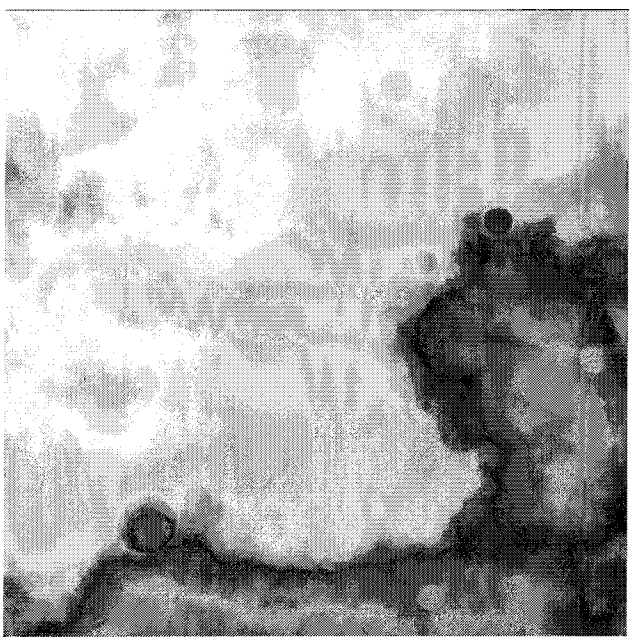

Figure 3 An example terrain map with craters and rocks.

3 Lidar Model

The simulation requires a model of a scanning LIDAR (LIght Detection And Ranging) so that realistic LIDAR measurements can be generated from synthetic terrain. This model should also be computationally efficient so that it can be used in real-time simulations. To this end an efficient high fidelity model that emulates the physical processes within a time-of-flight dual-axis gimbaled mirror LIDAR model has been created.

Principal of Operation

A LIDAR, when pointed at a surface, measures the range to this surface using laser light. A common range measurement principal used in long range LIDAR is time-of-flight; a laser pulse is emitted, the pulse is reflected from the surface, and the time of the returning pulse is recorded. This time is then converted to a range measurement using the speed of light. Because the laser beam has a finite divergence (angular width) the laser beam will be reflected from a patch of the surface and a continuum of ranges will result. The actual range measured by the LIDAR will be the range at which the integrated return light energy passes a threshold defined in the detector electronics. Usually this threshold is defined in terms of the signal to noise of the measurement.

A scanning LIDAR emits a continuous stream of laser pulses and optics are used to sweep the laser beam across a scene. Often single or dual axis gimbaled mirrors are used to direct the laser beam. In this case, a scanning LIDAR measurement, called a sample, consists of the detected range to the surface and the measured angular position of the mirror(s) when the laser pulse was emitted. Gimbaled encoders are not perfect, so noise will be introduced into the angle measurements.

A scan pattern defines the angular position of each sample. Modifiable parameters of a scan pattern are type of pattern (e.g., raster or spiral), the angular spacing between samples and the field of view of the scan. Using a model of the scanning LIDAR optical path, each sample is converted into a 3-D point in a Cartesian coordinate system attached to the LIDAR. The result is a set of 3-D point that convey that shape of the scene being scanned.

Measurement noise will be introduced into the 3-D points generated from LIDAR samples in multiple ways: range error from the integration of ranges within the footprint of the laser and range detector noise; mirror angle measurement noise; and LIDAR optical path model errors. A high fidelity model of a LIDAR should take all of these noise sources into account. Currently JPL is building a time-of-flight dual-axis gimbaled mirror LIDAR. For this reason, a model of such a sensor has been developed for the simulation described in this paper. First the efficient model used to generate range measurements given a lidar beam direction will be described. This will be followed by a description of the model used for laser beam scanning.

Range Measurement Model

Input into the range measurement model is a terrain map, a 3-D ray (origin and direction) describing the true

direction of the laser pulse impinging on the terrain, and the divergence of the laser pulse.

To account for the divergence of the laser, the pulse is modeled as bundle of rays centered on the pointing direction of the laser pulse. A portion of the energy in the laser pulse is given to each ray, and each of the rays is intersected with the terrain map (described below) to get a range. The energies of the rays are added in order of range (closest to farthest). When the cumulative energy exceeds a threshold, the range at that energy is the detected range. Detector noise is introduced by adding a gaussian distributed range error to the detected range and this is the output of the range measurement model.

The most computationally expensive portion of the lidar model is ray tracing: intersecting a ray with the synthetic surface to determine range. To alleviate this problem, an efficient ray tracing algorithm has been developed for ray/terrain map intersection. First the maximum z_{max} and minimum elevation z_{min} in the terrain map are determined before ray tracing. Then given a ray $\mathbf{r}(t)=(r_x(t),r_y(t),r_z(t))$ with origin $\mathbf{a} = (a_x,a_y,a_z)$ and direction $\mathbf{b}= (b_x,b_y,b_z)$, $\mathbf{r}(t) = \mathbf{a} + \mathbf{b}t$, t_{max} and t_{min}

$$t_{\max} = (z_{\max} - a_z)/b_z \quad t_{\min} = (z_{\min} - a_z)/b_z ,$$

that define the intersection of the ray with the maximum and minimum elevation planes, are computed.

Suppose the synthetic terrain map $T(r,c)$ has width H and sampling between grid cells of S then a 3-D point $\mathbf{x} = (x,y,z)$ can be projected vertically into the grid cell (r,c) in the terrain map using the operator $P(\mathbf{x})$

$$P(x) = P(x,y,z) = (r,c) = (y/S + H/2, x/S + H/2).$$

The intersection of the $\mathbf{r}(t)$ with the terrain map must occur along the line segment in the 2-D row/column space of the terrain map between

$$P(r(t_{\max})) = (r_{\max}, c_{\max}) \quad P(r(t_{\min})) = (r_{\min}, c_{\min}).$$

To find the intersection, this line segment and ray are searched starting at the grid cell $(r_s,c_s)= (r_{max},c_{max})$, and ray position $\mathbf{r}(t_s) = \mathbf{r}(t_{max})$ where $T(r_s, c_s) \le r_z(t_s)$, until the grid cell where $T(r_s, c_s) > r_z(t_s)$ is found. At this transition, the ray has passed though the terrain map, so the intersection has been found. Linear interpolation between the intersection and previous ray points and terrain elevations is used to get a sub-grid cell estimation of terrain map elevation z_i at the intersection. z_i is converted into a range ρ using

$$\rho = (z_i - a_z)/b_z$$

The procedure for ray intersection is used to get a range for all rays in the bundle describing the laser pulse. The energies associated with the rays are summed from closes to farthest and when the cumulative energy exceeds a threshold, that range is the error free range for the pulse ρ_n. Next, a gaussian error $G(\sigma_\rho)$ is added to this range to simulate detector noise. This range is then discretized to the resolution of the range detection electronics δ_ρ to get the range measurement ρ

Scanning Model

The scanning model is fairly simple because it models the pointing of the laser during scanning using a two-axis gimbaled mirror. The angles of the mirror are (θ,ϕ). The scan model has parameters for field of view f, number of samples within field of view n, and scan pattern. Currently the model only supports raster scans.

For a raster scan, the field of view and number of samples within the field of view define an array of equally spaced angles (θ,ϕ) that define the direction of the laser beam for each scan. Since pointing of the mirror is not perfect the true angles that the mirror is pointed to will not be exactly the same as those defined by the scan pattern. This effect is simulated in the scanning model by adding a gaussian distributed offset, $G(\sigma_\theta,\sigma_\phi)$ to each (θ_n,ϕ_n) pair to get $(\theta_{off},\phi_{off})$. $(\theta_{off},\phi_{off})$ and the position of the LIDAR during the scan, are then used to determine the range ρ to the surface using the range measurement model for each sample. To model sampling of angle measurements by the mirror encoders, (θ_n,ϕ_n) are discretized to the angular resolutions of the encoder $(\delta_\theta,\delta_\phi)$ to get the measured angles (θ,ϕ) The discretized angles and the measured range constitute the LIDAR sample (ρ,θ,ϕ) output for each pointing angle. The LIDAR samples for all angles in the scan patter constitute the output of the lidar model for a single scan.

In summary, the range measurement model has multiple parameters including the divergence of the laser pulse d, the number of rays used to model the pulse, the link analysis parameters of the laser beam used to set the threshold on return laser energy (laser power, pulse width, etc.), the detector error added to the measured range σ_ρ and the range resolution δ_ρ of the detector. The parameters input into the scanning model are field of view f, number of samples within field of view n, the pointing error distribution $G(\sigma_\theta,\sigma_\phi)$ and the mirror encoder sampling $(\delta_\theta,\delta_\phi)$.

Computational Issues

It takes less than 50μs to intersect a ray with a terrain map on a 174 MHz R10000 SGI O2 workstation. For a 100x100 sample scan with 1 ray per sample this takes 0.5 seconds which is sufficient for real–time simulation given that the JPL LIDAR under development is being designed to take a 10,000 sample scan in 1 second. With greater fidelity in the range measurement model (e.g. 10 rays per sample) a scan can be generated in 5 s. Obviously, greater fidelity cannot occur in real-time, but it may be useful during off-line Monte Carlo simulations.

4 Hazard Avoidance Algorithms

The inputs to hazard avoidance are lidar samples and a vector describing the geodetic normal of the surface, both in sensor coordinates. Also input is the safe landing parameters: lander diameter, maximum lander incidence angle and maximum surface roughness. The hazard avoidance algorithms process these inputs to select a safe landing site. Hazard avoidance takes part in three stages: terrain map generation, hazard detection and safe site selection.

4.1 Terrain Map Generation

Terrain map generation is the process by which range samples are projected into a grid to form a 2½-D terrain map representation (regridding). Scanning laser rangefinders generally have spherical or perspective projection models. Also, scan patterns are not always regular raster scans; spiral and triangle scans are common when minimizing scanner power. Nonlinear projection models and irregular scan patterns create an irregular sampling of the surface. If the range samples are used directly, a time-consuming hazard detection algorithm that accounts for the irregular spacing between samples is needed. However, by resampling the range samples from each scan to a regular grid in Cartesian space, hazard detection can be accomplished by applying fixed local operators to the resampled grid. Resampling greatly simplifies the underlying algorithms and data structures, so a more efficient algorithm results

A terrain map is a function $Z(r,c)$ that encodes elevation on a regular grid. To generate a terrain map, the horizontal size of each grid cell, s, and horizontal extent, h, of the terrain map must be determined. As shown in Figure 4, these parameters can be determined from the scanner field of view f, the average of scan samples across the scene n, and the average range to the scene being imaged R. In general we set these parameters as follows:

$$(1)\quad \begin{aligned} h &= 2R\tan(f/2) \\ s &= h/n \end{aligned}$$

With these settings, the terrain map will cover roughly the same extent as the scanned data and each grid cell will contain approximately one sample.

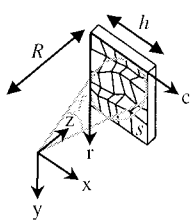

Figure 4 Sensor and terrain map coordinates.

Once the terrain map parameters are established, the procedure for terrain map generation is as follows. First, each range sample is converted from scanner angle and range coordinates to Cartesian coordinates (x,y,z). Next, the (x,y) coordinates of the sample are used to determine the floating point coordinates (r,c) that the sample projects to in the grid cell

$$(2)\quad (r,c) = (y/s + h/2,\ x/s + h/2)$$

The coordinate relationship between sensor and terrain map coordinates is shown in Figure 4. In general (r,c) will fall between discrete grid cells, so, to prevent aliasing, bilinear interpolation is used to update the terrain map. Two arrays are used to perform bilinear interpolation: the elevation accumulator $E(r,c)$ and the bilinear weight accumulator $W(r,c)$. For each sample, the four grid cells surrounding (r,c) are updated using

$$(3)\quad \begin{aligned} &p = r - \lfloor r \rfloor \quad q = c - \lfloor c \rfloor \\ &E(r,c) \mathrel{+}= (1-p)(1-q)z \quad W(r,c) \mathrel{+}= (1-p)(1-q) \\ &E(r+1,c) \mathrel{+}= p(1-q)z \quad W(r,c) \mathrel{+}= p(1-q) \\ &E(r,c+1) \mathrel{+}= (1-p)qz \quad W(r,c) \mathrel{+}= (1-p)q \\ &E(r+1,c+1) \mathrel{+}= pqz \quad W(r,c) \mathrel{+}= pq \end{aligned}$$

where $\lfloor\ \rfloor$ is the floor operator. After all samples have been accumulated, the elevation Z at each grid cell is determined using

$$(4)\quad Z(r,c) = E(r,c)/W(r,c)$$

Due to the irregular sampling by the scanner, it is possible that a grid cell did not have a sample projected into it and consequently does not have an elevation value. For efficiency during image alignment, it is important that the terrain map be free of holes, especially near the center of the map. A simple interpolation scheme is used to fill any holes. First, hole cells are detected by finding cells that do not have an elevation but are surrounded by cells with elevation. Next, each hole cell is assigned the average elevation of all neighboring cells that have elevation values. By repeating this process until all hole cells have an elevation value, the holes in the terrain map are filled incrementally. Figure 5 shows a typical range scan, a terrain map before hole filling and a terrain map after hole filling.

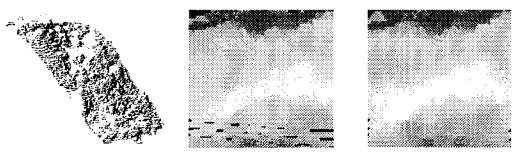

Figure 5 Terrain map generation.

4.2 Hazard Map Generation

The next step after generating the terrain map from the lidar samples is the generation of landing incidence angle and roughness maps that will be used in to select safe sites for landing. To achieve this, the surface must be separated into two components: the underlying smooth surface from which landing incidence angle is computed and the rocks and pits embedded in this surface from which surface roughness is computed. First the low frequency surface is computed; deviations from this surface are then determined by subtracting it from the original terrain map.

Robust Plane Fitting

Since the lander will be approaching the surface vertically, landing incidence angle is defined as the angle α between the local surface normal $\mathbf{n}$ and the geodetic normal of the surface $\mathbf{n}_g$. Local surface normal at a terrain grid cell is computed by fitting a plane to the terrain in the neighborhood of the cell; the computed local surface normal depends on the size of the neighborhood over which the plane parameters are estimated. For safe site selection, the normal of interest is the one that is computed with a patch that is the size of the lander footprint because this normal will indicate the resting angle between the lander and the surface. If the lander footprint is L meters then we approximate this size in units of grid cells as $l = \lceil L/s \rceil$ where $\lceil\ \rceil$ is the ceiling operator.

A plane is represented by the equation

$$\mathbf{n} \cdot \mathbf{x} + d = 0$$

where $\mathbf{n}$ is the surface normal of the plane and d is the plane intercept. The neighborhood of a grid cell can be represented by a set of 3-D points.

$$N = \{\mathbf{x}_1, \mathbf{x}_2 \ldots \mathbf{x}_m\}$$

Plane fitting finds the best fit plane ($\mathbf{n}$,d) to these points given some appropriate error metric. In many cases Mars terrain can be characterized by scattered rocks on top of relatively smooth terrain. The plane that is desired for estimating local surface normal is the plane fit to the surface in which the rocks are embedded. Given this domain specific insight, estimating the plane of the underlying smooth surface is a problem best solved by robust statistical methods where rocks are treated as outliers from the underlying smooth surface.

The robust statistical method used to fit a plane to the underlying terrain is Least Median Square (LMedSq) estimation. The LMedSq algorithm uses the following principal: three points define a plane; investigate multiple triples of 3-D points in N and eventually a triple that is free of outlier (rock) points will be found. The number of triples t that must be investigated is based on the expected percentage o of rock points in N and the desired probability P of obtaining a triple without outliers.

$$t = \ln(1-P)/\ln(1-(1-o)^3)$$

LMedSq estimation algorithms follow a standard form [6]; the LMedSq algorithm specialized to plane fitting is as follows. Repeat the following steps for t triples. First select a triple of non-collinear points ($\mathbf{x}_a$,$\mathbf{x}_b$,$\mathbf{x}_c$) randomly from N. Next compute the parameters ($\mathbf{n}$,d) for the plane that is defined by the three points using

$$\begin{aligned} \mathbf{n} &= (\mathbf{x}_b - \mathbf{x}_a) \times (\mathbf{x}_c - \mathbf{x}_a) \\ d &= -\mathbf{n} \cdot \mathbf{x}_a \end{aligned} \quad (5)$$

Given this plane, the square plane errors r_i are computed for each of the remaining point in N.

$$\{r_i\} = \{(\mathbf{n} \cdot \mathbf{x}_i + d)^2\}.$$

If the median of the above square plane errors is less than the median square plane error computed for all previous planes r_{med}, the current plane parameters becomes the best encountered so far $(\mathbf{n}_{best}, d_{best})$. The process is repeated for all of the t triples. Next the robust standard deviation

$$\sigma_r^2 = (1.4826(1+\frac{5}{t-3}))^2 r_{med}$$

is computed; using σ_r, a neighborhood point $\mathbf{x}_i$ is considered an outlier, and consequently eliminated, if

$$r_i = (\mathbf{n}_{best} \cdot \mathbf{x}_i + d_{best})^2 > \sigma_r$$

The final step in robust plane fitting is to fit a least-squares (LSq) plane to the remaining inlier points in the neighborhood. To fit a plane to multiple points we use the standard least squares plane-fitting algorithm based on finding the eigenvector of minimum eigenvalue of the scatter matrix of the points [2].

The LMedSq algorithm eliminates the points in the neighborhood that correspond to rocks and then fits a LSq plane to the remaining points. If a LSq plane was fit to the all of the points in the neighborhood, then the plane would be skewed in the direction of rock points, which would reduce the accuracy of the slope computation needed for hazard detection.

Map Generation

The robust plane($\mathbf{n}$,d) = (n_x,n_y,n_z,d) fit at a grid cell $\mathbf{x}$ = (x,y,z) intersects the underlying smooth rock free elevation z_s at the grid cell. z_s is defined using the robust plane parameters as

$$z_s = -(n_x x + n_y + d)/n_z \quad (6)$$

The robust plane also defines the landing incidence angle at the grid cell

(7) $\alpha = \cos^{-1}(\mathbf{n} \cdot \mathbf{n}_g / \|\mathbf{n}\| \|\mathbf{n}_g\|)$

If a robust plane is fit at every grid cell then the underlying smooth surface can be generated by computing z_s at every grid cell. Plane fitting is a relatively computationally expensive operation given the real-time requirements of landing, so computing a robust plane at every grid cell is infeasible. Furthermore, robust plane parameters will not change drastically between grid cells, so computing them at every cell is wasted effort.

To alleviate the computational burden, a robust plane is only fit to grid cells separated horizontally by the footprint of the lander l. Furthermore, the size of the neighborhood for each plane fit is set to a square regions centered around the grid cell of width l. The result is that robust planes are fit to a coarse grid of cells with non-overlapping neighborhoods that are the size of the lander. At each of these grid cells z_s and α are computed. In between grid cells bilinear interpolation, similar to that described in Section 4.1, is used to fill in the gaps between grid cells. The end result is are two maps $Z_s(r,c)$ and $A(r,c)$ that describe the underlying smooth surface and the landing incidence angle, respectively, at each grid cell in the terrain map.

Once the smooth surface has been generated, the roughness map of the surface $R(r,c)$ is simply computed as the difference between the smooth surface $Z_s(r,c)$ and the terrain map $Z(r,c)$. Only absolute deviations are needed to characterize roughness so

$$R(r,c) = |Z(r,c) - Z_s(r,c)|$$

4.3 Safe Site Selection

The strategy behind the safe site selection is to generate a landing cost map that first keeps the lander away from detected hazards and then, from the remaining terrain, selects the landing site that has minimal landing incidence angle and roughness.

Hazard map generation creates maps that define landing incidence angle and roughness at every grid cell in the terrain map. Using landing system constraints on maximum surface roughness R_{max} and landing incidence angle A_{max} safe sites in the terrain are selected as follows. First, the regions of the terrain that are hazardous to the lander are detected be identifying grid cells (r,c) where

(8) $R(r,c) > R_{\max} \quad or \quad A(r,c) > A_{\max}$

More specifically, a landing cost map $C(r,c)$ is constructed. If a grid cells (r,c) violates (8) then $C(r,c)$ is set to 1.0. Furthermore, the lander cannot intersect any grid cell that violates (8), so if (r,c) is within l of a grid cell that violates (8) then $C(r,c)$ is also set to 1.0. The remaining unassigned grid cells are assigned the normalized product of landing incidence angle and roughness.

$$C(r,c) = (R(r,c) * A(r,c)) / (R_{\max} * A_{\max})$$

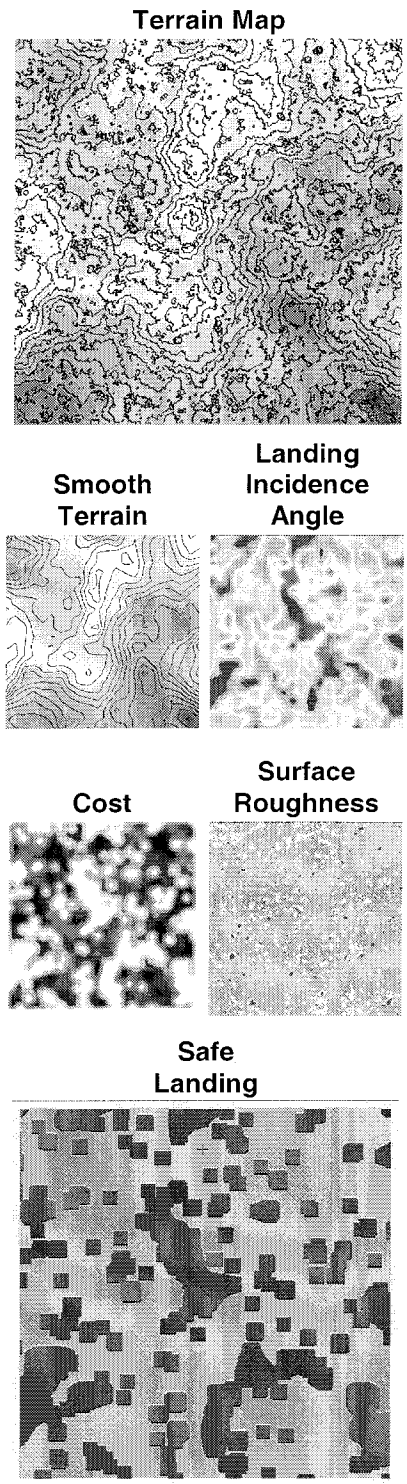

Figure 6 Hazard Detection Parameter Maps. Red indicates hazard, green indicates safe and yellow indicates unknown or partially safe.

$C(r,c)$ will be at or near 1.0 around grid cells that violate the landing constraints; $C(r,c)$ will be near zero for regions that have small roughness and landing incidence angle. By finding the minimum of $C(r,c)$, the best landing site will be found. However, in addition to minimizing the landing cost, the landing site selected should also be near regions of similarly low cost. This concept can be implemented by smoothing the cost map and then finding its minimum. Specifically, the cost map is smoothed by setting $C(r,c)$ to the average of all costs in a square neighborhood of size l centered around (r,c). The best safe landing site is then selected at the terrain coordinate (r,c) that minimizes the smoothed $C(r,c)$.

5 Powered Landing Guidance

Hazard-avoiding landing guidance was first developed for Apollo [1][5] and the guidance algorithm planned for Mars landings is a close cousin of Apollo's. In manned lunar landings, the commander identified landing hazards and manipulated a joy stick to redesignate the selected landing site to a hazard-free area of the lunar surface. In Mars landings, hazard detection and avoidance hardware and software take the place of the commander. Landing guidance cannot distinguish between human- and machine-derived landing-site redesignation commands. The most significant difference between landing on the moon and landing on Mars is that the Lunar Module approached the landing site along a nearly horizontal trajectory whereas the Mars lander will approach the site along a nearly vertical trajectory. This difference affects the guidance algorithm, as explained below.

Powered landing guidance transfers the lander from any current state (position and velocity) to touchdown in two phases, called "approach" and "vertical descent". Each phase has a target point. The approach-phase target may be five or ten m above the surface, and the vertical-descent target is near the surface, where the descent engines are shut down. As the lander approaches a target, it nulls horizontal components of velocity and acceleration so that it arrives at the target moving vertically downward in an erect attitude. The vertical descent is at constant velocity until the engines are shut off at or slightly before touchdown. Shutting off the engines before touching down makes the lander less likely to tip over.

The guidance algorithm solves a two-point boundary-value problem. The boundary values are the initial state and the target conditions. Target conditions are the position, velocity, and acceleration at the target point. By reaching zero horizontal components of target velocity and acceleration, the lander arrives at the target point moving vertically and erect.

To solve the two-point boundary-value problem, the guidance algorithm fits a polynomial between the current state and the target conditions. Thus the transfer trajectory is a polynomial in time. It expresses position and its derivatives velocity, acceleration, jerk, and snap as functions of the independent variable time.

With time (which is zero at the target and negative en route) denoted T, the position vector and its derivatives denoted R, V, A, J, and S, and current and target conditions denoted by subscripts C and T, the two-point boundary-value problem is expressed by

$$(9)\quad R_C = R_T + V_T T + A_T T^2/2 + J_T T^3/6 + S_T T^4/24$$

$$(10)\quad V_C = V_T + A_T T + J_T T^2/2 + S_T T^3/6$$

Both current vectors and the first three of the five target vectors are given. The solution is the following matrix equation, which expresses the target jerk and snap vectors in terms of the five given vectors.

$$(11)\quad \begin{bmatrix} J_T \\ S_T \end{bmatrix} = \begin{bmatrix} 24/T^3 & -6/T^2 \\ -72T^4 & 24/T^3 \end{bmatrix} \begin{bmatrix} R_C - R_T - V_T T - A_T T^2/2 \\ V_C - V_T - A_T T \end{bmatrix}$$

In terms of the target jerk and snap vectors, the current acceleration vector is given by

$$(12)\quad A_C = A_T + J_T T + S_T T^2/2$$

The guidance algorithm computes target jerk and snap until near the end of the phase, stopping short of the end to avoid dividing by powers of time as time approaches zero. It computes current acceleration until the end, using computed values of jerk and snap even when they are no longer being computed. The current acceleration is used for commanding engine thrust magnitude and direction.

This guidance algorithm solves the two-point boundary-value problem for any value of time. For Mars landings, time is chosen to produce a trajectory in which the vertical component of jerk is constant. This contrasts with Apollo guidance in which time was computed to produce a specified target value of the downrange component of jerk. In the Mars case, time is the solution of a quadratic equation, and in the Apollo case it was the solution of a cubic equation.

6 Simulation

The simulation ties together all of the previously described pieces. The simulation consists of three modules: the terrain server TS, the powered guidance module (PGM) and the lidar model/hazard avoidance module (HAM). The TS communicates with the HAM through TCP/IP sockets and the PGL and HAM communicate though named pipes.

First the terrain server is started with a command to generate a piece of terrain large enough to cover all LIDAR scans expected during the simulation. Next the PGM and HAM are started. The PGM starts with an

initial state and then propagates this state according to the powered guidance equations. There is no navigation or control errors in the simulation. At a rate of once per second, the PGM sends a command to the HAM to generate a LIDAR scan and detect the safest place to land. This command has the position and attitude of the spacecraft when the scan was requested and the current landing site. The HAM request some terrain from the terrain server, generates a LIDAR takes a scan and detects hazards. If the current landing site is hazardous then the HAM sends a new landing site back to PGM and the PGM computes the new trajectory to the new landing site. If the current landing site is safe then the HAM sends back the same current landing site and the PGM keeps the current trajectory. The Hazard Avoidance loop repeats until the spacecraft has landed. Parameters on landing site position, spacecraft trajectory and fuel consumption are recorded. If desired, images of hazard maps are recorder for each step of the simulation as well.

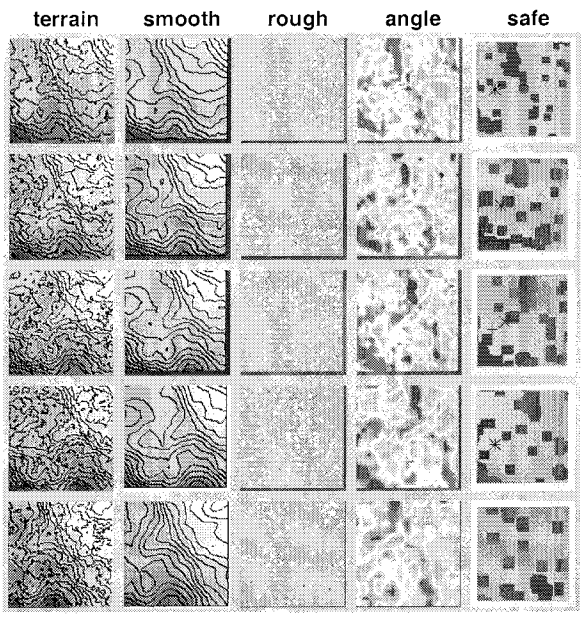

Figure 7 Example of simulation output. Each row of images corresponds to a single staep of the simulation. Note that there is a redesignation in the third row of images.

7 Simulation Results

To demonstrate the usefulness of the simulation, and prove the performance of the hazard avoidance algorithms, some Monte Carlo simulations were conducted. The first set of simulations quantifies the performance of the hazard detection and avoidance algorithms for a single LIDAR scan. The second set of simulations investigate the performance of the end to end powered landing system. The tests are preliminary and will be used to design more accurate and more robust guidance and hazard avoidance algorithms. The purpose of the tests is to demonstrate the variety of parameters that can be investigated with the simulation.

Hazard Avoidance

Three LIDAR parameters were investigated for there effect on hazard detection: LIDAR range detection error σ_ρ, LIDAR altitude a above the terrain and LIDAR resolution n. All other parameters were fixed. Figure 8 shows the results of the simulations in terms of safe landing site detection probability (i.e., the percentage of safe sites that were actually detected in the scan) and safe landing probability (i.e., the number of landing sites selected by the hazard avoidance algorithm that were actually safe when compared to ground truth).

The performance of hazard avoidance with respect to altitude is poor for very low altitude then there is an improvement at moderate altitudes and finally again at high altitude the performance drops off. The poor performance at low altitude is caused by terrains used in the simulation some terrain having no safe place to land; all of the terrain is hazardous. This observation leads indicates that it better to make hazard avoidance decisions when high above the surface. Because the LIDAR to laser footprint increases with altitude, hazard detection performance degrades with altitude. This is the reason for decreasing hazard avoidance performance at high altitude.

The performance of hazard detection with respect to scanning resolution is as expected. As scanning resolution increases, the horizontal spacing between samples on the surface decreases. This will result in more hazards being detected and consequently better hazard avoidance performance.

Hazard avoidance performance decreases as range detector noise increases. As detector noise increases, more false hazards and safe zones will be detected, so the probability of choosing a hazardous landing site increase.

Safe Landing

Safe landing probability for a complete run of the landing simulation (multiple scans and redesignations during descent) was investigated as a function of rock density. Multiple Monte Carlo landings were performed and the number of times that the final landing site was safe, when compared to ground truth, was measured. The results are given in Table 1.

Table 1 Safe landing simulation performance as a function of rock density.

	Rock Density
	0.1
Average Deviation From Landing Site (m)	50
Safe Landing Probability	0.98

8 Conclusions

The safe landing simulator described in this paper will be used during design of the next generation of Mars landers and will eventually be incorporated into a complete 12 degree of freedom simulation of all phases of Martian entry, descent and landing. Types of questions the simulation will help answer are:

- What are the requirements for the LIDAR in terms of field of view and maximum range?
- How much fuel is consumed during landing?
- What control authority is needed to land safely given
- certain terrain statistics?
- What is the probability of safe landing?

References

[1] G.W. Cherry, "E Guidance - A General Explicit, Optimizing Guidance Law for Rocket-Propelled Spacecraft", MIT Instrumentation Laboratory (now C.S. Draper Laboratory) Report R-456, 1964.

[2] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*, Wiley-Interscience, New York, 1973.

[3] R. Gaskell, "Martian Surface Simulations," Jour. Geophysical Research-Planets, 98(E6), pp. 11099-11103, 1993.

[4] R. Gaskell, J. Collier, L. Husman and R. Chen, "Synthetic Terrain Environments for Simulated Missions," JPL Internal Report, 2001.

[5] A.R. Klumpp, "Apollo Lunar Descent Guidance", *Automatica*, Vol 10, pp. 133-146, Pergamon Press, 1974.

[6] C. Stewart, "Robust Parameter Estimation in Computer Vision," SIAM Review 41(3), pp. 513-537, 1999.

Acknowledgements

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

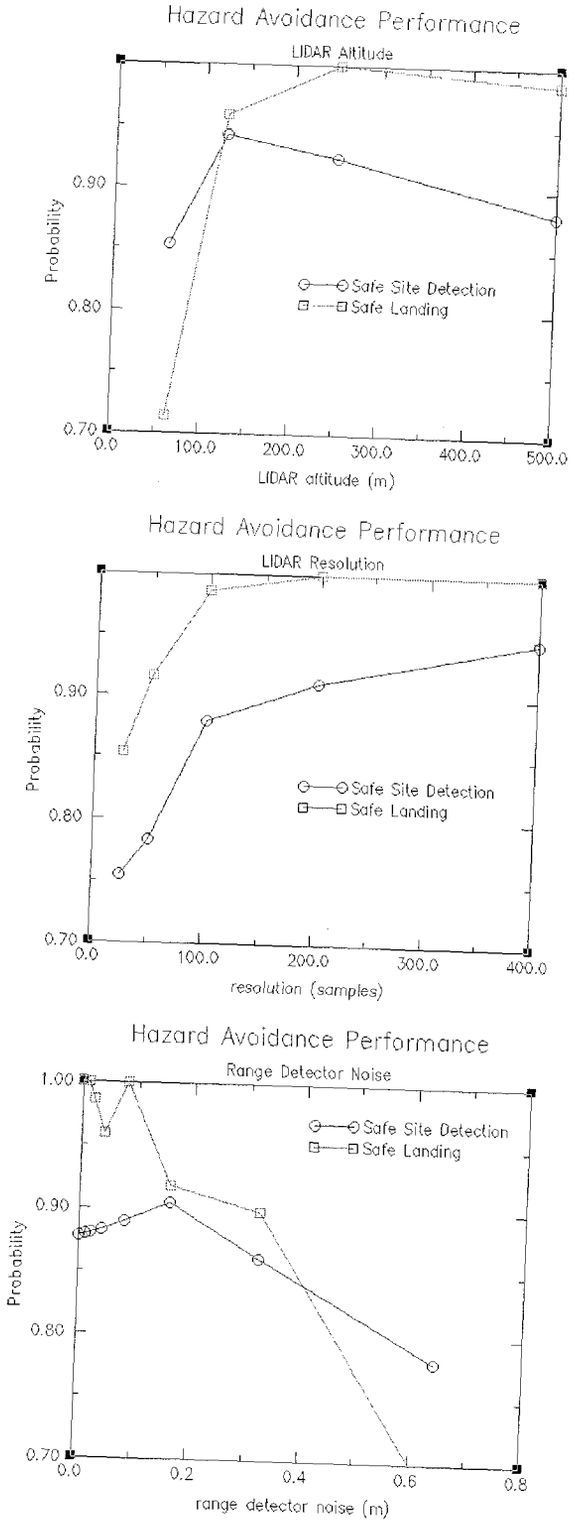

Figure 8 Hazard avoidance performance.